LA-UR- 02-0945

Approved for public release;
distribution is unlimited.

Title: Quality Improvement of Surface Triangulations Using Local Parameterization

Author(s): Rao V. Garimella, Mikhail Shashkov
Mathematical Modeling and Analysis (T-7),
Los Alamos National Laboratory, Los Alamos, NM, USA.

Patrick M. Knupp
Parallel Computing Sciences Department,
Sandia National Laboratories, Albuquerque, NM, USA.

Submitted to: 8th International Conference on Numerical Grid Generation on Computational Field Simulations,
June 2-6, Honolulu, Hawaii, USA

Los Alamos
NATIONAL LABORATORY

Quality Improvement of Surface Triangulations Using Local Parameterization[†]

Rao V. Garimella, Mikhail Shashkov
Mathematical Modeling and Analysis (T-7), Theoretical Division,
Los Alamos National Laboratory, Los Alamos, NM, USA.
E-mail: rao@lanl.gov, shashkov@lanl.gov

Patrick M. Knupp
Parallel Computing Sciences Department,
Sandia National Laboratories, Albuquerque, NM, USA.
E-mail: pknupp@sandia.gov

Abstract

A procedure has been developed for improvement of surface triangulation quality by node repositioning as directed by a two-stage non-linear optimization technique. The surface mesh modification procedure is designed so that the quality of the triangles is improved without drastically distorting the surface. Repositioning of the nodes is done using local parametric mapping so that they remain on the triangles of the original mesh.

Introduction

Surface meshes play an important role in mesh generation and mesh based analysis applications. The success of mesh generation algorithms like the advancing front algorithm and the quality of solid elements they generate depends on the quality of the surface mesh. Surface meshes also define boundaries of computational domains, and therefore, their quality can strongly influence the accuracy of numerical analysis procedures. Therefore, optimization of surface mesh quality is a very important problem. This paper presents a method for improving the quality of a surface triangulation by node repositioning.

An important consideration in surface mesh optimization is to minimize the deviation of the modified mesh from the original mesh while improving element quality. When the surface mesh is constructed upon an underlying smooth surface, it is usual to use the 2D parametric space of the smooth surface to reposition nodes. Repositioning nodes in the parametric space

[†]LA-UR-02-0945

of the surface guarantees that the nodes will remain on the surface when they are mapped back to real space. Then, the modified and original meshes approximate the surface equally well, considering the deviations of the mesh triangles from the smooth surface.

Ensuring the similarity between the original and modified meshes is more challenging when there is no underlying smooth surface for reference. Such surface triangulations occur frequently during surface reconstruction using sampled data points. Also, in moving mesh simulations, triangulations based on a smooth surface may get deformed enough so that the parameterization of the surface is unusable for node repositioning. One method employed for surface optimization in such cases is to compute a parametric map from the surface triangulation and use it to reposition points of the mesh [1]. Doing so ensures that the nodes of the mesh stay on the triangles of the original mesh. However, developing a global parameterization of a triangulation requires a computationally expensive solution of a non-linear system of equations. Also, trianglulations of closed surfaces must be cut into two or more pieces and mapped as separate surfaces with this method.

In this paper, a procedure is presented to optimize surface meshes by repositioning nodes in a series of local parametric spaces constructed by barycentric mapping of triangles [2] of the original or *base* mesh. The repositioning of nodes is directed by a numerical optimization procedure [3] that is designed to improve the geometric shape of triangles while keeping the modified mesh as close as possible to the base mesh. The method has been tested on a number of surface meshes and has proved to be very effective.

Overview

The surface quality optimization procedure described here consists of two stages. The first stage is a local optimization process in which the optimal position of each mesh vertex is calculated with respect to the fixed positions of its neighbors. The objective function of the local minimization is constructed from Jacobian matrix condition numbers of triangles connected to the vertex under consideration [3, 4].

Consider a vertex V_i, connected to a set of of edges, $\{E(V_i)\}$, and triangles, $\{F(V_i)\}$. Assume that one of the triangles $F_j \in \{F(V_i)\}$ has edges $E_p \in \{E(V_i)\}$ and $E_q \in \{E(V_i)\}$ connected to vertex V_i. Then, the *Jacobian matrix*, $\mathbf{J}_{ji}$, of F_j at vertex V_i is defined as $\mathbf{J}_{ji} = [\mathbf{e}_p \; \mathbf{e}_q]$ where, $\mathbf{e}_p$ and $\mathbf{e}_q$ are edge vectors representing edges E_p and E_q of lengths l_p and l_q respectively.

Since $\mathbf{J}_{ji}$ is a 3x2 matrix for a 3D triangle, its condition number must be calculated by singular value decomposition methods. On the other hand, the Jacobian matrix of a triangle in 2D space is a 2x2 matrix whose condition number can be calculated more easily as $\kappa(\mathbf{J}_{ji}) = (l_p^2 + l_q^2)/A_j$, where A_j is twice the area of face F_j [3, 4]. This condition number is only a function of triangle lengths[1]; therefore, it is invariant with rotation of the triangle in the plane. Since there always exists a coordinate system in which an arbitrarily oriented triangle lies on one of its coordinate planes, it suggests that the condition number is also useful for measuring the quality of arbitrarily oriented triangles in space.

Using the definition of Jacobian matrices for triangles, an objective function for the first stage optimization is defined as:

$$\psi^c(\mathbf{x}_i) = \sum_j \kappa(\mathbf{J}_{ji}(\mathbf{x}_i)) = \sum_j \frac{l_p^2(\mathbf{x}_i) + l_q^2(\mathbf{x}_i)}{A_j(\mathbf{x}_i)}, \qquad j \in \{j \mid F_j \in \{F(V_i)\}\}$$

where l_p and l_q are defined as before and $\mathbf{x}_i$ is the coordinate vector of V_i. Note the presence of area A_j in the denominator as a barrier function which discourages node movements that tend to make the triangle degenerate.

The vertex position calculated in the first optimization is stored as the *reference position* of the vertex but the vertex is not moved to this location. After reference positions are calculated for all mesh vertices, two *reference edge vectors* are calculated for each edge in the mesh; each reference edge vector goes from the reference position of one vertex of the edge to the original position of the other. The idea of reference edges is illustrated in Figure 1, where E_k is an edge with vertices V_i and V_j. The reference positions of V_i and V_j are V_i^R and V_j^R respectively. The two reference edges vectors for E_k are $(\mathbf{e}_k^R)_i$ and $(\mathbf{e}_k^R)_j$, where the outer subscript indicates which of the vertices is at its reference position.

Using the concept of reference edge vectors, it is now possible to define *Reference Jacobian Matrices* just as Jacobian matrices were defined for a mesh without reference positions. Therefore, if the edges of F_j connected to vertex V_i are E_p and E_q, then the reference Jacobian of F_j at V_i is defined as $\mathbf{J}_{ji}^R = \left[(\mathbf{e}_p^R)_i \; (\mathbf{e}_q^R)_i\right]$.

The second stage of the surface mesh optimization is a global optimization procedure based on the definition of reference Jacobian matrices. Its goal is to find a configuration for all the mesh edges such that a compromise is struck between the various pairs of reference edge vectors while forming a valid mesh with improved triangle quality. Since the reference edge vectors

[1] A_j can be expressed as a function of the lengths of the triangle sides

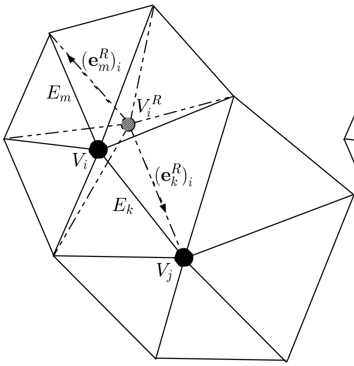


Figure 1. Reference positions and reference edge vectors.

were formed by locally improving the quality of triangles connected to each mesh vertex, it can be expected that a valid set of edge vectors that achieves a compromise between the various reference edge vectors will also improve the quality of the mesh. Also, since each reference edge vector has one of its vertices at the location of the original or base mesh, it is expected that the optimized mesh will not deviate drastically from the base mesh.

To achieve the goal of the global optimization, an objective function is formed that minimizes the difference between the Jacobian matrices of the mesh being optimized and the reference Jacobian matrices as shown below:

$$\Psi^R = \sum_i \sum_j \frac{\|\mathbf{J}_{ji} - \mathbf{J}_{ji}^R\|^2}{A_j / A_{ji}^R}, \; i \in \{i \mid V_i \in \{V\}\}, \; j \in \{j \mid F_j \in \{F(V_i)\}\}$$

where, $\{V\}$ is the set of all mesh vertices, $||.||$ is the Frobenius norm, A_{ji}^R is the twice the reference triangle area, i.e., the area of F_j with vertex V_i at the reference position V_i^R. Note that, similar to the objective function for local optimization, the objective function includes a barrier term A_j in the denominator in the form of the triangle area to prevent mesh invalidity.

It is desirable to use a global procedure in the minimization of Ψ^R, in which all the mesh vertices are moved simultaneously. However, for surface optimization with local parametrization, the parametric bounds impose too strong a constraint on a global optimization process. The line search in a global optimization seeks a single step size for the parametric coordinates of all the vertices in the mesh. Even if a parametric coordinate for a single vertex goes out of bounds, the line search must end for all the parameters in the problem, making the optimization very inefficient.

The solution adopted here is to reposition the vertices one at a time using a local piece of the global objective function. Consider a vertex V_i, connected to the set of faces $\{F(V_i)\}$. Then the piece of the global objective function that involves the real or reference position of V_i is given as:

$$\psi_i^R = \sum_j \sum_k \frac{\|J_{jk} - J_{jk}^R\|^2}{A_j / A_{jk}^R},\ j \in \{j \mid F_j \in \{F(V_i)\}\},\ k \in \{k \mid V_k \in \{V(F_j)\}\}$$

In the expression, the outer sum is over all triangles connected to the vertex and the inner sum is over all vertices of a triangle. Each term in the summation refers to the k^{th} vertex of the j^{th} triangle, F_j, connected to V_i.

With this modification, the two stages of the surface mesh optimization become similar since both involve calculating an optimal position for each vertex by minimizing a local function with respect to parametric coordinates. For the first stage or local optimization stage, one iteration is made over vertices of the mesh and the optimal but virtual position of each vertex is calculated with its neighboring vertices fixed at their original positions. The optimal positions are then used to calculate reference edge vectors for the second stage. In the second stage or global optimization by local iteration, several iterations are made over the mesh to reposition the vertices. During each iteration, the optimal position of each vertex is calculated taking into account the positions of its neighboring vertices calculated in a previous iteration. The iterations end when the movement of all the vertices is negligible. At that point all the vertex positions are updated to the newly computed positions resulting in an optimized surface mesh.

Optimization with respect to Parametric Coordinates

The local objective functions, ψ^c and ψ^R, defined in Section 2 are in terms of real coordinates of the vertices. If an optimization procedure is applied directly to these objective functions, it may indicate vertex movement off the base surface mesh. To constrain the movement of the vertices to the triangles of the base mesh, the objective functions are optimized with respect to a local mapping of the triangles into 2D space.

Each base mesh triangle is parameterized using a barycentric mapping [2], giving rise to parametric coordinates $0 \leq (s_1, s_2) \leq 1$. All objective function evaluations are performed after transforming the parametric coordinates into real coordinates using the barycentric mapping. The gradient of the objective function with respect to the parametric coordinates is calculated using numerical differentiation. The gradient direction is used to compute a search direction, $\mathbf{d}$, in parametric space according to the principles of the non-linear conjugate gradient method [5].

Vertices on surface boundaries (model edges) are also moved using a local mapping of mesh edges that are on these boundaries. Each of these mesh edges can easily be mapped into parametric space using one parameter $0 \leq s_1 \leq 1$. Then the gradient of the objective function[2] with respect to this one parameter is calculated and line search direction is determined to be along or opposite to the edge direction. Vertices that are at corners (model vertices) or vertices that are fixed by the user are not moved.

Line Search or 1D minimization

The purpose of the line search is to find a distance, α, along the parametric search direction, $\mathbf{d}$, such that the objective function is minimized or the constraints of the line search are encountered. For surface optimization with local parameterization, the line search is subject to two constraints, parametric bounds and mesh validity. During the line search, if the parametric bounds of the base triangle mapping are reached, the point has reached an edge of the triangle in real space. Proceeding any further along that direction will move the point out of the base triangle and off the triangulation. For example, in Figure 2a, the line search tries to proceed from point 2 to 3' (which is outside the triangle and off the surface triangulation) but encounters the parametric bounds of the triangle at 3. Also, it is possible that one of the triangles connected to the vertex becomes invalid due to the movement along the search direction in which case the line search must be terminated. This is shown in Figure 2b where the line search must be terminated at point 2 because further movement towards point 2' renders the shaded triangle invalid.

The line search procedure is implemented as an incremental stepping algorithm with step size control. The line search starts with a very small step size and checks if the function has decreased, the parameters are within bounds and if the mesh is valid. If so, the step size is increased and the process is repeated; if not, the step size is cut in half (up to a minimum) and the checks are repeated. The algorithm has additional refinements for zeroing in on the minimum with better accuracy.

[2] The form of the objective function remains the same no matter where it is computed at.

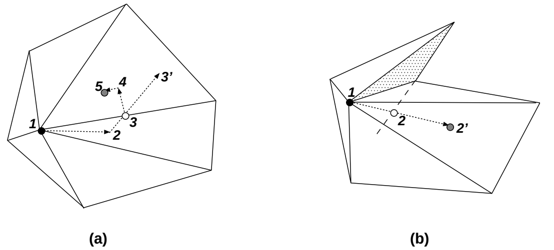


Figure 2. Line search constraints (a) Parameter bounds (b) Invalid Mesh.

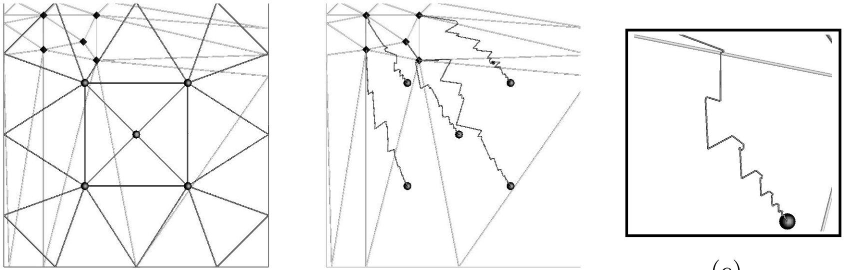
Parameter Update and Parametrization Change

Once the line search along a direction has terminated, the step size (α) obtained from it is used to update the parametric coordinates of the vertex as $\mathbf{s}_{new} = \mathbf{s}_{old} + \alpha\mathbf{d}$. If the line search terminated normally at a minimum or because further movement in the search direction would have made the mesh invalid, the optimization iterations are continued as usual with a new gradient calculation. However, if the line search terminated because the parametric bounds were reached, then it is assumed that the vertex is trying to move out of the current base triangle. In such a case, the optimization is terminated, the adjacent triangle is adopted as the base triangle and the optimization restarted in the parametric space of the new triangle. If the vertex is at an edge of the base mesh and flips too many times between the adjacent triangles, it is taken to be an indication that the line search must proceed along the edge. The line search direction along the edge is taken to be the one closer to the negative of the gradient direction.

Figure 3 illustrates the movement of vertices during the optimization with respect to parametric coordinates using a planar triangulation example. The mesh was optimized by performing several iterations of local optimizations using the condition number function, ψ^c, over all the vertices.

Results

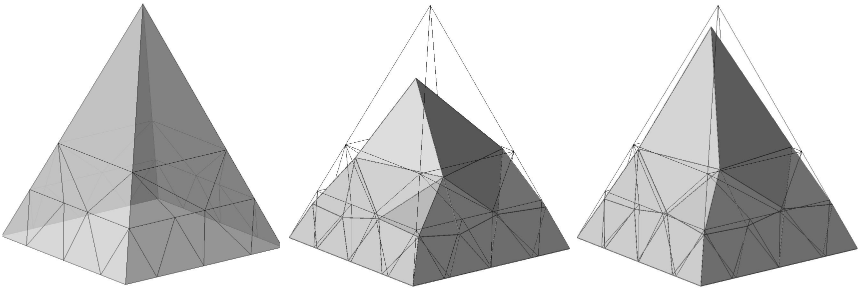
Figure 4 shows a simple example to illustrate the effect of a global condition number optimization and reference Jacobian based optimization on a non-planar surface mesh. The objective function of the global condition number optimization is defined as sum of the local condition numbers at all mesh vertices. Figure 4a shows the original pyramid shaped mesh on which the two optimization techniques are applied. Figure 4b shows the effect of optimizing globally using just a Jacobian matrix condition number based objective function. Figure 4c shows the effect of optimizing the mesh using



(a) (b) (c)

Figure 3. (a) Original (light lines) and final (dark lines) mesh (b) Paths taken by vertices to their final positions (c) Zoom-in of one of the paths.

the two-stage optimization method described in this paper. Clearly, the two-stage optimization method using reference Jacobians causes less deviation from the original mesh than the global condition number optimization, even though it does not improve the shape of the triangles as much.

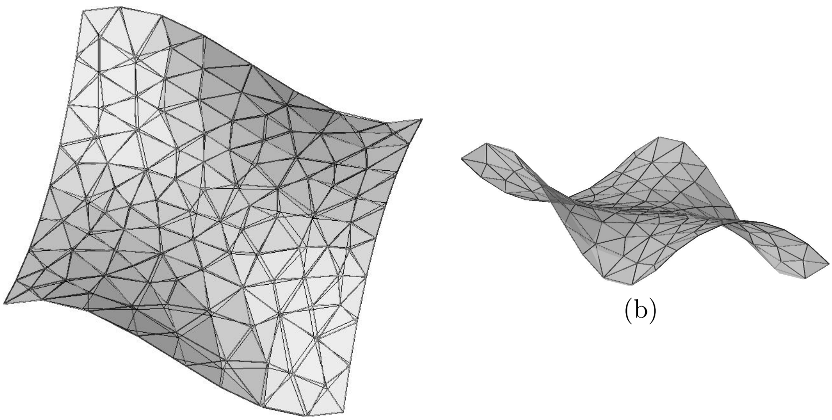


(a) (b) (c)

Figure 4. (a) Original Mesh (b) Mesh optimized with condition number objective function (c) Optimized with reference jacobian objective function.

Figure 5 shows two views of an optimized triangulation superimposed on the original curved surface triangulation. In Figure 5(a), the light lines show the edges of the original mesh and the dark lines show the optimized mesh. From the picture it is clear that the shape of the triangles is improved but optimization process has not distorted the original surface much.

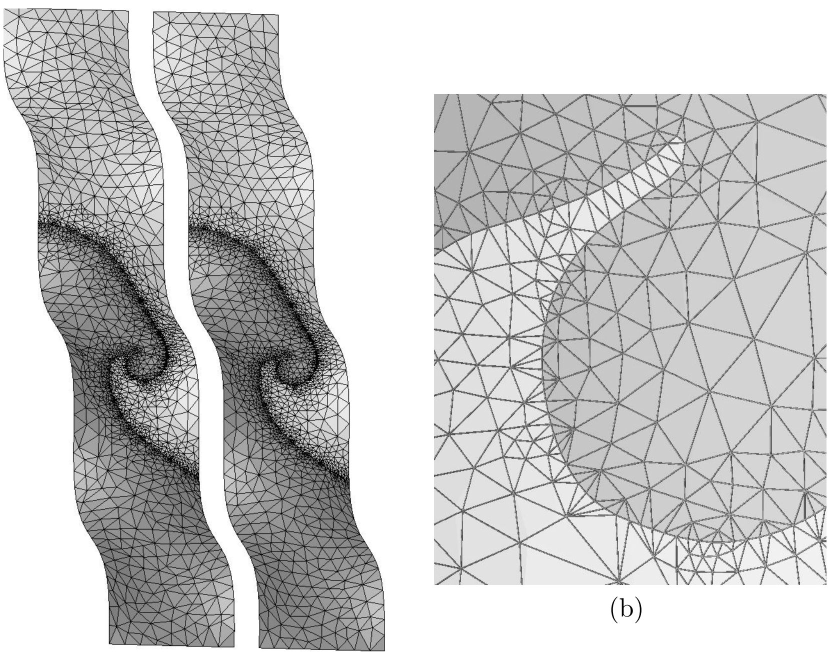
Figure 6 shows an example of a complex surface mesh improved with the surface optimization procedure. The original mesh was constructed by taking a slice of a Raleigh-Taylor simulation mesh and curving it. Note that the triangulation consists of two surfaces sharing a common interface. Figure 6a shows the original and improved meshes and Figure 6b shows a



(b)

(a)

Figure 5. Superimposed views of optimized and original mesh (light edges) for curved surface



(b)

(a)

Figure 6. (a) Original (left) and Final (right) meshes of multi-domain curved surface triangulation (b) Zoom-in of optimized mesh edges overlaid on original mesh triangles

zoom-in of the overlaid meshes. In Figure 6b, the edges of the optimized mesh have been superimposed on the triangles of the original mesh. The figures show that the method improves the triangle quality, and preserves the shape of the surface and the curved boundaries of the surface.

Conclusions

A procedure was presented to improve the quality of complex surface meshes using numerical optimization. The optimization is designed to improve the quality of the triangles without distorting the discrete surface too much. The vertices are kept on the original surface triangulation with the help of a barycentric mapping of triangles. The procedure has been successfully tested on a number of complex surface meshes. Future work will extend the procedure to quadrilateral meshes and also develop more quantitative measures for distortion of the surface.

Acknowledgements

This work was performed at Los Alamos National Laboratory operated by the University of California for the US Department of Energy under contract W-7405-ENG-36. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not edorse the viewpoint of a publication or guarantee its technical correctness.

The authors also acknowledge use of software tools from the Scientific Computation Research Center, Rensselaer Polytechnic Institute, Troy, NY that enabled the testing of the ideas presented in this paper.

References

[1] A Sheffer and E de Sturler. Parameterization of faceted surfaces for meshing using angle-based flattening. *ENGINEERING WITH COMPUTERS*, 17(3):326–337, 2001.

[2] T. J. R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis.* Prentice Hall, 1987.

[3] M. Shashkov and P.M. Knupp. Optimization-based reference-matrix rezone strategies for arbitrary lagrangian-eulerian methods on unstructured grids. In *Proceedings of the 10th Anniversary International Meshing Roundtable*, pages 167–176, Newport Beach, CA, 2001. Sandia National Laboratories.

[4] P.M. Knupp. Achieving finite element mesh quality via optimization of the jacobian matrix norm and associated quantitites. Part I - a framework for surface mesh optimization. *International Journal for Numerical Methods in Engineering*, (48):401–420, 2000.

[5] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, 1999.